САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

факультет Прикладной Математики – Процессов Управления

Кафедра Механики Управляемого Движения

Голованова Алина Владимировна

Выпускная квалификационная работа бакалавра

Разработка системы управления траекторным движением мобильного робота

Научный руководитель:

к.ф. - м.н., доцент Шиманчук Д. В.

Санкт-Петербург

2017 г.

Содержание

1 Введение

В настоящее время робототехника активно развивается, все больше и больше задач становятся доступны для выполнения роботами. Робот – машина, созданная по принципу живого организма, предназначенная для выполнения некоторых действий, управляемая заранее заданной программой и получающая информацию о состоянии окружающей среды с помощью датчиков. Люди используют роботов для того, чтобы облегчить свой труд и не тратить время на выполнение монотонной, не требующей принятия решений работы. Также роботы используются в тех областях, где существует угроза жизни человека – при исследовании космоса, под водой, при пожарах.

RRT (rapidly exploring random tree) – это алгоритм, разработанный для эффективного поиска путей в пространстве путем построения случайного дерева, заполненяющего пространство. Быстро исследующие случайные деревья – динамическая структура данных, предназначенная для исследования пространства состояний. Дерево строится постепенно из ребер и вершин, выбранных случайным образом из пространства поиска, и по своей природе его рост имеет случайный характер. RRT - алгоритмы были разработаны Steven M. LaValle и James J. Kuffner Jr [1, 3]. Эти алгоритмы хорошо работают

не только со стационарными препятствиями, но и динамическими, и используются при автономном планировании движения робота.

2 Постановка задачи и обзор литературы

2.1 Постановка задачи

Задача ставится следующим образом: требуется определить такое движение робота, которое обеспечивает его переход из заданного начального положения, в заданное конечное положение. Робот должен достичь конечное положение, проходя мимо препятствий не задевая их. Для решения исходной задачи будем решать следующие две подзадачи:

- Глобальную – нахождение абсолютных координат полюса робота – планирование траектории. Траектория выбирается еще до начала движения на основе имеющейся карты.

- Локальную – разработка управляемой математической модели робота, на основании которой может быть произведена оценка управляющих воздействий, которые обеспечивают последовательное достижение найденных характеристических точек траектории. На заданном этапе управления для определенной характеристической точки определяется часть модельной траектории робота, в конечной точке которой осуществляется следующий этам движения и построение следующей части модельной траектории. Процесс заканчивается при попадании робота в конечную характеристическую точку траектории.

2.2 Обзор литературы

В данной работе основная задача решается с помощью RRT-алгоритма. Алгоритмы, использующие быстро исследующие случайные деревья были разработаны и описаны Steven M. LaValle и James J. Kuffner Jr [1, 3]. Статья подробно описывает роботу алгоритма. Также авторы приводят примеры его использования.

Для решения локальной задачи используется метод тангенциального избегания, описанный в работе Ferreira, Pereira, Vassallo, Filho and Filho в 2008г [2].

3 Глава 1. Решение глобальной задачи

3.1 Описание математической модели

Для решения глобальной проблемы была поставлена задача планирования траектории мобильного робота на карте с препятствиями (нахождения точек траектории движения робота).

Имеется поле и координаты старта и финиша робота. так же мы знаем размеры робота (радиус). Необходимо построить такую траекторию, чтобы при движении по ней робот из старта достиг финишной точки, и не задевал препятствия во время движения.

3.2 Реализация

Для нахождения точек используется алгоритм RRT.

```
Algorithm BuildRRT
  Input: Initial configuration qinit,
  number of vertices in RRT K,
  incremental distance q )
  Output: RRT graph G

  G.init(qinit)
  for k = 1 to K
    qrand      RAND_CONF()
    qnear      NEAREST_VERTEX(qrand, G)
    qnew       NEW_CONF(qnear, qrand, q )
    G.add_vertex(qnew)
    G.add_edge(qnear, qnew)
  return G
```

Этот алгоритм исследует пространство, начиная из начального состояния, пока не достигнет необходимого конечного состояния. В первую очередь производим дискретизацию, заданная область разбивается сеткой 1000×1000. На каждом шаге алгоритм выбирает слу-

чайную точку из из получившейся сетки с помощью функции nextRand():

```
function nextRand() {
        var next = {};
        var temp_x = 0;
        var temp_y = 0;
         while (Math.abs(temp_x) < 0.01) {
            temp_x = 1000*Math.random();
        }

        while(Math.abs(temp_y) < 0.01) {
            temp_y = 1000*Math.random();
        }

        next.x = Math.round(temp_x);
        next.y = Math.round(temp_y);

        return next;
    };
```

Далее алгоритм находит ближайшую из точек-вершин уже построенного дерева, проверяя также, чтобы добавленное ребро не имело пересечения с препятствиями.

```
while (distance(tree[cnt], finish) > err) {
        cnt++;
        do {
            t = nextRand();
            temp.x = (1.0 * height * t.x)/1000;
            temp.y = (1.0 * width * t.y )/1000;
            best = -1;
            dist = 999999;
           for(i = 0; i < cnt; i++) {
                if (checkIntersection(temp.x,
                temp.y, tree[i].x, tree[i].y)
                && distance(temp, tree[i]) < dist)
                {
                    dist = distance(temp, tree[i]);
                    best = i;
                }
            }
        } while (best < 0);
        draw.line(tree[best].x, tree[best].y,
        temp.x, temp.y).stroke({ color: '#000',
```

```
        width: 1, linecap: 'round' });
        point.x = temp.x;
        point.y = temp.y;
        point.prev = best;
        tree[cnt] = point;
    }
```

Визуализируем полученные данные с помощью фреймворка SVG.js:

```
var tmp = tree[cnt].prev;

    while(tree[tmp].prev != -1) {
        alert(tmp);
        draw.line(tree[tmp].x,
        tree[tmp].y,
        tree[tree[tmp].prev].x,
        tree[tree[tmp].prev].y)
        .stroke({ color: '#ff0',
        width: 1, linecap: 'round' });
        tmp = tree[tmp].prev;
    }
```

3.3 Результаты

Разработано веб-приложение на языке JavaScript. При открытии страницы пользователю предлагается форма, куда можно ввести начальные данные (координаты старта, финиша, радиус), а так же координаты препятствий. При нажатии кнопки draw программа рисует карту, а так же дерево, полученное с помощью алгоритма rrt. Поддерево, содержащее полученные точки траектории выделено желтым цветом. Изображение карты и пути можно сохранить в формате png. Полученные на этом этапе точки будут использованы при решении локальной задаче в следующей главе.

4 Глава 2. Решение локальной задачи

4.1 Разработка управляемой математической модели

Робот с дифференциальным приводом имеет 2 колеса. Каждое его колесо имеет свой мотор. Это позволяет колесом вращаться с разной скоростью, за счет чего возможно изменение направления во время двжения

• Если колеса вращаются с одинаковыми скоростями, робот движется прямолинейно.

• Для разворота на месте необходимо установить скорости, одинаковые по модулю, но имеющие разное напривление.

• В других случаях – движение по дуге.

Обозначим скорости колес (линейные скорости с которыми они «покрывают» поверхность) V_L и V_R - для левого и правого колес, соответственно, и W расстояние между колесами.

• Прямолинейное движение, если $V_L = V_R$

• Разворот на месте, если $V_L = -V_R$

• В остальных случаях робот будет двигаться по дуге.

Для нахождения радиуса R криволинейного пути, рассмотрим период движения Δt , в течении которого робот движется вдоль дуги окружности, имеющей угол $\Delta\theta$.

• Левое колесо: пройденное расстояние = $V_L\Delta\theta$; радиус дуги = $R - \frac{W}{2}$

• Правое колесо: пройденное расстояние = $V_R\Delta\theta$; радиус дуги = $R - \frac{W}{2}$

• Обе колесные дуги имеют в основании одинаковый угол $\Delta\theta$

$\Delta\theta = \frac{V_L\Delta t}{R-\frac{W}{2}} = \frac{V_R\Delta t}{R+\frac{W}{2}}$

$\Rightarrow \frac{W}{2}(V_L + V_R) = R(V_R - V_L)$

$\Rightarrow R = \frac{W(V_L+V_R)}{2(V_R-V_L)}, \Delta\theta = \frac{(V_R-V_L)\Delta t}{W}$

Рассматривается движение робота в среде с препятствиями.

ρ – расстояние до точки P_i,

θ – угол между осью Ox и направлением на точку P_i (азимут),

ϕ – угол курса робота,

α – курсовой угол, разность между курсом и азимутом,

v – линейная скорость полюса мобильного робота.

Управление роботом осуществляется заданием двух независимых управляемых колес. Кинематическая модель имеет следующее математическое описание:

$$\begin{cases} \dot{x} = v\cos\phi, \\ \dot{y} = v\sin\phi, \\ \dot{\phi} = \frac{v_2-v_1}{R}, \end{cases} \tag{1}$$

где

$v = \frac{v_1+v_2}{2}$,

$v_1 = w_1 \cdot r$,

$v_2 = w_2 \cdot r$,

x, y – проекции скоростей полюса робота на абсолютные оси

r – радиус колес

R – расстояние между колесами

v_1, v_2 – линейные скорости осей колес робота

w_1, w_2 – угловые скорости вращения колес

Робот должен достигнуть последовательно каждую точку траектории P_i $(\rho_i \to 0, \alpha_i \to 0)$, $i = 0, \ldots, n$.

Определим математическую модель, которая описывает навигацию робота при движении к точке P_i в полярных координатах:

$$\begin{cases} v_r = \dot{\rho} = -v\cos\alpha, \\ v_\theta = \rho\dot{\theta} = -v\sin\alpha, \\ \dot{\alpha} = \dot{\phi} - \dot{\theta}. \end{cases} \Rightarrow \begin{cases} \dot{\rho} = -v\cos\alpha, \\ \dot{\theta} = -\frac{v\sin\alpha}{\rho}, \\ \dot{\alpha} = \dot{\phi} + \frac{v\sin\alpha}{\rho}. \end{cases} \quad (2)$$

Таким образом получаем, что робот может управляться значениями линейной и угловой скоростей - v, w. Для этого в работе (2008г) предлагается воспользоваться математический аппаратом теории управления - построение функции Ляпунова

$V(\rho, \alpha) = \frac{\rho^2}{2} + \frac{\alpha^2}{2}$

где ее производная в силу системы (1) должна быть неположительна, то обеспечит невозрастание расстояния до точки P_i и значения курсового угла α.

$$\dot{V}(\rho, \alpha) = \rho \cdot \dot{\rho} + \alpha \cdot \dot{\alpha} = -\rho v \cos\alpha + \alpha(w + \frac{v \sin\alpha}{\rho}). \tag{3}$$

Полная производная по времени от функции Ляпунова в силу системы (2) будет отрицательно-определенная, если в качестве значений управляющих воздействий выбрать следующие значения для управляющих скоростей:

$$\begin{cases} v = v_m ax \cdot \operatorname{th}\rho \cdot \cos\alpha, \\ w = -k_w \cdot \alpha - v_m ax \cdot \frac{\operatorname{th}\rho}{\rho} \cdot \sin\alpha\cos\alpha, k_w > 0. \end{cases} \tag{4}$$

5 Выводы

Представленное исследование может быть использовано для создания реального робототехнического комплекса, который способен на основании полученной карты местности и идентифицированным на ней препятствиям производить построение траектории с дальнейшим ее использованием системой управления траекторным движением мобильного робота, проект которого также представлен в работе.

6 Заключение

В результате проделанной работы было:

1. Рассмотрены модификации алгоритмов быстро-исследующих случайных деревьев (RRT-алгоритм) для определения точек возможной траектории движения полюса мобильного робота.

2. Реализован RRT-алгоритм, который находит точки траектории, обеспечивающей обход заданных стационарных препятствий при перемещении мобильного робота из начального положения в конечное.

3. В приложении представлен исходный код программы, которая при задании конфигурации поля (размеры поля, радиус робота, координаты препятствий) позволяет произвести моделирование траектории полюса робота. Примеры работы алгоритма представлены на рисунках

4. Представлена математическая модель управляемого движения мобильного робота с дифференциальным приводом, на основании реализации которой можно провести оценку управляющих воздействий, обеспечивающих построение модельной траектории основания мобильного робота.

Список литературы

[1] LaValle, Steven M «Rapidly-exploring random trees: A new tool for path planning ». Technical Report. Computer Science Department, Iowa State University (TR 98-11).

[2] André Ferreira; Flávio Garcia Pereira; Raquel Frizera Vassallo; Teodiano Freire Bastos Filho; Mário Sarcinelli Filho «An approach to avoid obstacles in mobile robot navigation: the tangential escape ». Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal do Espírito Santo - Av. Fernando Ferrari, 514 - 29.075-910 Vitória, ES, Brazil

[3] LaValle, Steven M, James J Kuffner Jr. «Randomized Kinodynamic Planning ».

[4] Fahad Islam, Jauwairia Nasir, Usman Malik, Yasar Ayaz and Osman Hasan «RRT*-Smart: Rapid convergence implementation of RRT* towards optimal solution «